AMENDMENTS TO THE CLAIMS:

Please cancel without prejudice claims 35-37 and 39-42 and amend claims 1-4, 6-11, 13-17, 19-24, 26-29 and 31-34 as follows.

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (currently amended) A method of generating software test information, said method comprising the steps of:

a) generating, from a sequence of original instructions, at least one original instruction of said sequence of original instructions including~~of which includes~~ a condition code, a corresponding sequence of generated instructions, wherein for selected original instructions having a~~said~~ condition code, ~~wherein~~ a corresponding generated instruction is a predetermined generated instruction having a generated opcode including said~~corresponding~~ condition code;

b) executing, on a target processor, said corresponding sequence of generated instructions and thereby producing software test information; and

c) when during said step (b) said predetermined generated instruction is encountered, determining using a check performed in hardware by said target processor upon said condition code within said generated opcode with reference to status information associated with an operation of said target processor whether the corresponding condition code of said predetermined generated instruction is satisfied and, if so, replacing said predetermined generated instruction with said corresponding original instruction from said sequence of original instructions so as to cause said corresponding original instruction to be executed, wherein said

predetermined generated instruction is an instruction which is not recognised by said target processor.

2. (currently amended) The method of claim 1, wherein each instruction of said sequence of original instructions includes a condition code.

3. (currently amended) The method of claim 1, wherein said condition code is an instruction qualifier which prevents said original~~the~~ instruction from being executed by said target processor unless said status information satisfies said condition code.

4. (currently amended) The method of claim 1, wherein said status information is predetermined architectural state associated with said target processor and said condition code specifies a status of said predetermined architectural state that must be met in order for ~~the~~said original instruction to be executed.

5. (cancelled).

6. (currently amended) The method of claim 1, wherein said step a) comprises the step of:

generating, from said sequence of original instructions, a corresponding sequence of generated instructions, a predetermined generated instruction being generated for each instruction in the sequence of original instructions.

7. (currently amended) The method of claim 1, wherein said step a) comprises the steps of:

a1) partitioning said sequence of original instructions into a number of instruction groups, each instruction group including one or more original instructions; and

a2) generating said predetermined generated instruction for one original instruction in each of said instruction groups.

8. (currently amended) The method of claim 7, wherein said step a2) comprises the step of:

generating said predetermined generated instruction for ~~the~~a last original instruction in each of said instruction groups.

9. (currently amended) The method of claim 7, wherein said predetermined generated instruction provides information relating to the number of original instructions in a corresponding instruction group of said number of instruction groups.

10. (currently amended) The method of claim 1, wherein said step c) further comprises the step of:

incrementing a coverage counter when the condition code of the predetermined generated instruction is satisfied to provide an indication that said corresponding original instruction will be executed.

11. (currently amended) The method of claim 1, wherein said step c) further comprises the step of:

incrementing a counter associated with said corresponding original instruction when the condition code of the predetermined generated instruction is satisfied to provide an indication that said corresponding original instruction will be executed.

12. (original) The method of claim 11, wherein said step c) further comprises the step of:

replacing a preceding instruction in said sequence of generated instructions with said predetermined generated instruction having a condition code corresponding to said preceding instruction.

13. (currently amended) The method of claim 1, wherein said step c) further comprises the step of:

executing said corresponding original instruction on said target processor.

14. (currently amended) An apparatus for generating software test information, said apparatus comprising:

instruction generation logic configured to generate, from a sequence of original instructions, at least one original instruction of said sequence of original instructions ~~which~~ and includes a condition code, a corresponding sequence of generated instructions, wherein, for selected original instructions having ~~a~~said condition code, ~~wherein~~ a corresponding generated instruction is a predetermined generated instruction having a generated opcode including said~~a~~ ~~corresponding~~ condition code;

a target processor configured to execute said corresponding sequence of generated instructions thereby producing software test information and to identify the occurrence of said predetermined generated instructions; and

determination logic configured, when said predetermined generated instruction is encountered by said target processor, to determine using a check performed in hardware by said target processor upon said condition code within said generated opcode with reference to status information associated with an operation of said target processor whether the corresponding condition code of said predetermined generated instruction is satisfied and, if so, to replace said predetermined generated instruction with said corresponding original instruction from said sequence of original instructions so as to cause said corresponding original instruction to be executed, wherein said predetermined generated instruction is an instruction which is not recognised by said target processor.

15. (currently amended) The apparatus of claim 14, wherein each instruction of said sequence of original instructions includes a condition code.

16. (currently amended) The apparatus of claim 14, wherein said condition code is an instruction qualifier which prevents ~~the~~said original instruction from being executed by said target processor unless said status information satisfies said condition code.

17. (currently amended) The apparatus of claim 14, wherein said status information is predetermined architectural state associated with said target processor and said condition code

specifies a status of said predetermined architectural state that must be met in order for ~~the~~said original instruction to be executed.

18. (cancelled).

19. (currently amended) The apparatus of claim 14, wherein said instruction generation logic is operable to generate, from said sequence of original instructions, a corresponding sequence of generated instructions, a predetermined generated instruction being generated for each instruction in the sequence of original instructions.

20. (currently amended) The apparatus of claim 14, wherein said instruction generation logic is operable to partition said sequence of original instructions into a number of instruction groups, each instruction group including one or more original instructions, and to generate said predetermined generated instruction for one original instruction in each of said instruction groups.

21. (currently amended) The apparatus of claim 20, wherein said instruction generation logic is operable to generate said predetermined generated instruction for ~~the~~a last original instruction in each of said instruction groups.

22. (currently amended) The apparatus of claim 20, wherein said predetermined generated instruction provides information relating to the number of original instructions in a corresponding instruction group of said number of instruction groups.

23. (currently amended) The apparatus of claim 14, wherein said determination logic is operable to increment a coverage counter when the condition code of the predetermined generated instruction is satisfied to provide an indication that said corresponding original instruction will be executed.

24. (currently amended) The apparatus of claim 14, wherein said determination logic is operable to increment a counter associated with said corresponding original instruction when the condition code of the predetermined generated instruction is satisfied to provide an indication that said corresponding original instruction will be executed.

25. (original) The apparatus of claim 24, wherein said determination logic is operable to replace a preceding instruction in said sequence of generated instructions with said predetermined generated instruction having a condition code corresponding to said preceding instruction.

26. (currently amended) The apparatus of claim 14, wherein said determination logic is operable to cause the execution of said corresponding original instruction on said target processor.

27. (currently amended) A computer program product comprising a computer readable storage medium containing computer readable instructions that, when executed on a computer, generate software test instructions for testing a target processor by performing the step of:

a) generating, from a sequence of original instructions, at least one original instruction of said sequence of original instructions including ~~which includes~~ a condition code, a corresponding sequence of generated instructions as said software test instructions, wherein for selected original instructions having ~~a~~said condition code, ~~wherein~~ a corresponding generated instruction is a predetermined generated instruction having a generated opcode including said ~~a corresponding~~ condition code, wherein said predetermined generated instruction is an instruction which is not recognised by ~~a~~said target processor.

28. (currently amended) The computer program product of claim 27, wherein each instruction of said sequence of original instructions includes a condition code.

29. (currently amended) The computer program product of claim 27, wherein said condition code is an instruction qualifier which prevents ~~the~~said original instruction from being executed by a target processor unless said status information satisfies said condition code.

30. (cancelled).

31. (currently amended) The computer program product of claim 27, wherein said step a) comprises the step of:

generating, from said sequence of instructions, a sequence of generated instructions, a predetermined generated instruction being generated for each instruction in the sequence of original instructions.

32. (currently amended) The computer program product of claim 27, wherein said step a) comprises the steps of:

a1) partitioning said sequence of original instructions into a number of instruction groups, each instruction group including one or more original instructions; and

a2) generating said predetermined generated instruction for one original instruction in each of said instruction groups.

33. (currently amended) The computer program product of claim 32, wherein said step a2) comprises the step of:

generating said predetermined generated instruction for ~~the~~a last original instruction in each of said instruction groups.

34. (currently amended) The computer program product of claim 32, wherein said predetermined generated instruction provides information relating to the number of original instructions in a corresponding instruction group of said number of instruction groups.

35. (cancelled).

36. (cancelled).

37. (cancelled).

38. (cancelled).

39. (cancelled).

40. (cancelled).

41. (cancelled).

42. (cancelled).